Explicit Intelligence in Adaptive Hypermedia: Generic Adaptation Languages for Learning Preferences and Styles

Natalia Stash
Faculty of Computer Science and Mathematics
Eindhoven University of Technology
Postbus 513, 5600 MB Eindhoven, The Netherlands
+31-40-247 3874
nstach@win.tue.nl

Alexandra Cristea
Faculty of Computer Science and Mathematics
Eindhoven University of Technology
Postbus 513, 5600 MB Eindhoven, The Netherlands
+31-40-247 4350
a.i.cristea@tue.nl

Paul De Bra
Faculty of Computer Science and Mathematics
Eindhoven University of Technology
Postbus 513, 5600 MB Eindhoven, The Netherlands
+31-40-247 2733
debra@win.tue.nl

ABSTRACT
This paper deals with a new challenge in Adaptive Hypermedia and Web-based systems: finding the perfect *adaptation language* to express, independently from the domain model (or even platform), the *intelligent, adaptive behaviour of personalized Web courseware*. The major requirements for the ideal language are: *reuse*, *flexibility*, *high level semantics*, and *ease of use*. To draw closer to this ideal language, we compare two such language proposals: LAG, a generic adaptation language, and a new XML adaptation language for *learning styles* in AHA!, LAG-XLS.

Categories and Subject Descriptors
H.1 [**Information Systems**] Models and Principles; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia - *architectures, navigation, user issues*; H.3.3 [**Data**]: Data Structures - *distributed data structures, graphs and networks*; K.3.1 [**Computers and Education**]: Computer Uses in Education - *distance learning.*

General Terms
Design, Human Factors, Standardization, Languages, Theory.

Keywords
Plug-and-play intelligence, learning styles, user modelling, adaptive hypermedia, authoring of adaptive hypermedia.

1. ADAPTATION LANGUAGE AS AN INTERMEDIATE PLATFORM

1.1 Benefits on an Intermediate Platform
Creation and authoring of adaptive, "intelligent" courseware can be a cumbersome process [13]. In order to create a personalized, rich learning experience for each user, not only the actual content of the lesson has to be prepared, but much more. Catering for different user needs means creating different (labelled) alternatives of the same content; this in turn leads to multiple paths through that content. This content organization is often called in the Adaptive Hypermedia (AH) literature the creation of the *Domain Model* [42]. Adaptive dynamics design also embraces the specification concerning the kind of user expected, given the content alternatives. Often, this is done in the form of user attributes that are specified in what is usually called a *User Model* [6] (UM). Moreover, in the educational field there is a serious need for a separate *Pedagogical Model* [14], which establishes adaptation and interaction types for the different kind of learners, according to pedagogic strategies [9]. Finally, machine specifications and constraints have to be considered, via, e.g., a *Presentation Model* [13]. All these are connected via an *Adaptation Model* [42]. Therefore, authoring of personalized courseware can be a difficult and costly process. There are different ways of striving towards alleviating what can be called the 'authoring problem'. Two ways of dealing with it are:

1. To consider the difficulty of the first-time authoring process unavoidable and to concentrate on improving *reuse* capabilities. In this way, the cost could be reduced by reuse of previously created material and other components.
2. To lighten the authoring burden, by moving away from the platform dependent authoring style, common especially in adaptive hypermedia web-based systems, towards *platform independent* authoring.

In this paper we concentrate on the first solution, *reuse*. The solving of one issue actually offers solutions of the second, but in this paper, this issue is not followed-up. This paper looks instead at what we consider the most challenging and difficult task for the reuse topic: reuse of *intelligent, adaptive behaviour*. The approach taken here is to look at *defining Adaptation Languages*, as vehicles for the intelligent behaviour of the AH. This means that the reusable items are not only the static parts of the authored courseware (such as the content of the DM), but the actual dynamics as well. This would be equivalent to *exchanging not only the ingredients, but the recipes as well*. This extraction and separate expression of semantically relevant, reusable, explicit 'artificial intelligence' of AH systems can also feedback to AH

Conference'05, Month 1–2, 2005, City, State, Country.

systems as analyzer of the level of 'intelligence' they can provide. As requirements we enforce that the adaptation behaviour described by the adaptation languages should be reusable and the language extensible –the latter as it may be necessary to be able to create new strategies that need the addition of new elements. Ideally, the language should use or extend the emerging Web standards, which will enhance reusability and compatibility with current implementations.

Next, we are going to shortly describe the connection of our research to standards, especially from the point of view of static element reuse.

1.2 Connection to standards

Here we look at what current standards apply or are connected to recent AH research[1] in general and can be used or *extended* for our research, in particular. Firstly, we are aiming at a model-driven architecture [30], having as a goal the separation of the model from its implementation, for enhancing flexibility and reuse. Secondly, as we aim at educational systems [20], we can follow the Information Technology Standard bodies (ISO, JTC1, IEC), informed by CEN and IEEE[2]. Specifically, we aim at reuse of both *static* and *dynamic* elements. E-learning standards enabling *reuse* (please note however that it is mainly reuse of static material, and not dynamic, as in our research), are: *Learning resources*: metadata: IEEE-LOM (Learning Object Metadata) [22], Dublin Core (Metadata for Electronic Resources) [18], ADL-SCORM [3] (Reusable learning content as "instructional objects"); *Data exchange*: IMS-CP (Content Packaging) IMS-CPS, *Data formats*: IMS-QTI (Question and Test Interoperability), *Education Modelling Languages* (EML [19]), e.g. with learning paths specifications: IMS-LD (Learning Design [24]). From the point of view of defining the users of the learning systems, we have: *Learner model* IEEE-PAPI (Public and Private Information for Learners) [21], IMS-LIP (Learner Information Package Specification) [23]; *Accessibility*: Web Accessibility Guidelines (WAI); *Rights*: Creative Commons. This long list of standards can specifically influence AH educational research. A careful selection is needed to avoid clashes resulting from the use of more standards together.

Recently, within AH Web-based systems, such standards started being used for adaptive learning systems (e.g., Personal Reader [17]; ALE [27])[1]. Even if AH requirements are not exactly met by these standards, implementations based on them try to keep as much as possible of the AH capabilities. Similarly, adaptive hypermedia techniques are starting to be applied to the heavily standardized semantic web [40] developments [10].

The future might also see more elements of the dynamics of adaptive systems entering the standards field. We believe this is the only way to ensure real interfacing, exchange and reuse. One attempt in this direction is the IMS ADL Simple Sequencing Protocol [25]. However, this protocol fails to provide all the features of adaptation that adaptive hypermedia can offer [1], such as disconnecting the prerequisites from the domain model. Therefore, at present, more research has to be done to determine what the *best representation for such adaptive, intelligent courseware dynamics* is, and here is where our research fits in.

Next, we are going to briefly analyse the current state of the art concerning how and what type of system dynamics and adaptation is implemented in current courseware.

1.3 Elements of Course Dynamics in AH

In order to extract the main elements of personalized, intelligent course dynamics, we work on a concrete case of representing personalization based on learning styles (LS). This choice is based on the degree of difficulty of representing LS-based strategies. LS and their effects on learning have been examined most carefully in [9]. This 182 page report "reviews the literature on learning styles and examines in detail thirteen of the most influential models. The report concludes that it matters fundamentally which instrument is chosen. The implications for teaching and learning in post-16 learning are serious and should be of concern to learners, teachers and trainers, managers, researchers and inspectors."

Our review of research on and application of learning styles in Adaptive Hypermedia [34] shows that existing systems can provide adaptation to the learner in terms of *content adaptation* [6], *navigation paths* [6] or usage of *multiple navigational tools* [16] These adaptation types limit the possible response of the system to accommodate the different learning styles. The most frequently used elements of *instructional strategies* [9] we have found in Adaptive Web-based Education literature are:

- Selection of media items to accommodate different learner preferences; this can also be extended to different learning styles. For instance, the same information (or the same concept) can be presented in various ways, by using alternative media types [6]– audio, video, image, text, etc. Depending on the learner's style a certain item (or group of items) may be included into the final presentation. In terms of learning styles, we can say that the *verbalizers* [32], who prefer textual information, may be presented with text and possibly spoken audio; whilst the *imagers*, who prefer pictorial information, can be shown images, diagrams, graphs, charts or other items about the same concept [32]. The selection process can be applied not only to media items, but also to other types of items.
- Ordering information or providing different navigation paths. The order in which information items are processed can be based on learner needs. For instance, some learners prefer to learn things by *doing something actively* first whilst others prefer to *collect data first* and then turn to action (this corresponds to the *active* and *reflective* learning styles, respectively). Moreover, some learners tend to learn through a linear, step-by-step process which is logical and systematic, whilst others want to see the big picture before they tackle the details (this corresponds, respectively, to the *sequential* and *global* learning styles [20]).
- Providing learners with navigational support tools. Depending on the learner preferences, different learning tools can be provided. In terms of catering for learning styles, for example, *field-dependent* [41] learners can be provided with a concept map, graphic path indicator, advanced organizer, etc. in order to help them organize the structure of the knowledge domain. Alternatively, *field-independent* learners might be provided with a control option showing a menu from which they can choose to proceed with the application in any order [37].

[1] Traditionally, AH research ignored standards.

[2] LTSC: Learning Technol. Standards Committee; CEN updated by Prometeus & Ariadne [4]; IEEE by IMS, AICC & ADL

There are fewer systems which attempt to provide, along with various instructional strategies, some mechanisms for *inferring* the learner's preferences based on his/her actions and selections. For example, MANIC [36] uses a Naïve Bayes Classifier to reason about the learner's preferences in terms of explanations, examples and graphics; in iWeaver [26] the application of a Bayesian network is planned to predict and recommend media representations to the learner.

1.4 A View on Adaptive Learning Strategies

To distinguish between the different types of strategies, we need, beside the previous list of elements of instructional strategies, a high-level classification based on their overall semantics. From the analysis of the literature, we observe that we can classify strategies according to their *application range*, as follows:

- *Instructional strategies* – define how the adaptation is performed. Namely the adaptation rules specified in the strategy are used to adjust the presentation to the learner with a particular learning preference [26], style [36] or need [6]. We argue that it is very important to provide several instructional strategies for an application so that the learners or tutors can try different ones and select the most appropriate.
- *Instructional meta-strategies – inference* or *monitoring strategies* – are applied in order to infer the learner's preferences during his/her interaction with the system [36],[34]. These strategies cannot completely replace the existing psychological questionnaires for determining learning styles; however they can be used as a simplified, unobtrusive way to infer the learner preferences corresponding to these styles via their browsing behaviour.

The first type of strategy is more frequently used, but the second type is still novel and requires some clarification. A meta-strategy can for example, track the learner's preferences by observing his/her interactions with the system [36]. It can track some repetitive patterns in the learner's behaviour, like accessing particular types of information (if a choice of different types is available). It can observe that the user has a preference for textual information, which is typical for a learner with *verbalizer* style, or, on the contrary, that the user has a preference for the pictorial representations (*imagers* or *visualizers)*. It can also trace the navigational paths: browsing through the learning material in breadth-first order - typical for the learners with *field-dependent* or *holist* style - versus navigating in depth-first order, that might indicate a learner with *analytic* style [9],[31]. Meta-strategies of this type update some user model parameters which can be used later on for selecting a particular instructional strategy. These parameters can indicate what the system 'thinks' the learner's preferences are. In most existing systems that provide adaptation to a learner's styles, information about the learning styles and preferences is not updated during the interaction. However, the learning style preferences might actually change, depending on various circumstances [9] (for instance on the mood, time of day, subject, etc.). Meta-strategies could trace if the preferences specified by the learner when he begins working with the system stay the same or change. In case the learner's behaviour is different than initially specified, a strategy corresponding to another learning style might be suggested. Other examples of user model parameters which can be influenced by the actions specified in the meta-strategies are: level of difficulty of the material presented to the learner, link colours, etc. These actions occur when the learner accesses the concepts of an application.

According to the type of adaptation provided, we can refine the classification of adaptation strategies by analyzing the external (interactive) *actions* occurring, as follows (see Table 1).

Table 1. Refined classification of actions in adaptive strategies.

Basic actions on items	Selection Showing the content of an item Showing a link to an item
Hierarchical actions on items	Actions on child items Actions on parent item
Actions on groups of items (e.g., siblings)	Ordering Performing 'actions on items' on each group item
Actions on the *overall environment*	Changing the layout of the presentation

These are actions which directly determine changes in what the users sees. Similar to meta-strategies, instructional strategies also perform internal actions (mainly user model updates). These actions can be classified according to traditional user model classification and are therefore not further explained here.

In the following, we will show how we have used our analysis of the state of the art and of the standards as well as an existing adaptation language, LAG, to create a new language, based on Web technologies.

2. LAG: Model, Language & Implementation

The LAG model is a specification of the Adaptation Model, as defined by the LAOS model [13]. LAOS is a generic model for authoring of adaptive hypermedia, detailing a *Domain Model*, a *User Model*, and a *Goal and Constraints Model* (which becomes the *Pedagogical Model* for educational applications), a *Presentation Model* (dealing with the different machine-oriented ways of presenting the same information: e.g., different colours, formats, etc.) and an *Adaptation Model*. For the purpose of this paper, we only focus on the Adaptation Model, the sub-model that allows reuse of dynamics, as opposed to current standards which are mainly focused on static material reuse. The Adaptation Model is the one representing the Artificial Intelligence component of the Adaptive System.

2.1 LAG Model and Language: Review

In order to enable reuse of dynamics in personalization and adaptation, we transformed the adaptation model into a 3-layer model, LAG [12]. LAG consists, at the lowest level, of an *Adaptation Assembly Language*, corresponding to the typical IF-THEN rules in adaptive hypermedia. At the intermediate level, the model requires a semantic *Adaptation Language*. This language should incorporate more semantics, which should allow reuse in different learning situations. Finally, at the highest level, the LAG model situates *Adaptive Strategies* or *Adaptive Procedures*. These strategies/ procedures[3] are containers for the actual adaptation program (which details, in machine readable adaptation language, how the adaptation will be performed). In addition each strategy has a description (semantic label) in natural language, which can be directly used by authors to select a specific, ready-made

[3] Procedures are new language constructs extending the language.

strategy for their course. In this way, course contents creation and the creation of adaptation dynamics for that course are kept separate, and can be performed by different authors, at different times (or by the same author at different times).

As an instantiation of the Adaptation Language in the LAG model, the LAG Language [12] was introduced. This language uses for syntax the LAG grammar (Figure 1), and is the basis of an Intermediate Platform specification for adaptation dynamics. Concretely, the LAG Language provides the building blocks for the creation of Adaptation Strategies. Figure 1 shows the new, extended version of the LAG grammar, improved after authoring usability tests [11], as well as conversion and reuse tests (into two delivery systems, AHA! [2] and WHURLE [28]).

```
PROG → DESCRIPTION VARS
       INITIALIZATION  IMPLEMENTATION
DESCRIPTION → "// DESCRIPTION" COMMENT
VARS → ATTRIBUTE | (VARS)+ "," VARS
INITIALIZATION → "initialization("
                    STATEMENT ")"
IMPLEMENTATION →"implementation("STATEMENT ")"
STATEMENT → IFSTAT | WHILESTAT | FORSTAT |
  BREAKSTAT | GENSTAT |  SPECSTAT | COMMENT |
  (STATEMENT) * STATEMENT | ACTION
IFSTAT → if CONDITION then (STATEMENT)
WHILESTAT → while CONDITION do
            (STATEMENT) [TARGETLABEL]
FORSTAT → for RANGE do (STATEMENT)
          [TARGETLABEL]
BREAKSTAT → break SOURCELABEL
GENSTAT → generalize((CONDITION)*)
SPECSTAT → specialize((CONDITION)*)
ACTION → ATTRIBUTE OP VALUE
COMMENT → "//""text" | (COMMENT)* COMMENT
CONDITION → enough((PREREQ)+, VALUE) | PREREQ
RANGE → "integer"
PREREQ → ATTRIBUTE COMPARE VALUE
LABEL → "text"
TARGETLABEL → "text"
SOURCELABEL → "text_label_a"
ATTRIBUTE → GENCONCEPT | SPECCONCEPT
GENCONCEPT → "CM_type.concept.attr" |
             "CM_type.concept.attr_z"
SPECCONCEPT → "CM_x.concept_y.attr_z"
OP → "=" | "+=" | "-=" | ".="
COMPARE → "==" | "<" | ">" | "in"
VALUE → "text"
```

Figure 1. The extrended LAG Grammar.

The figure describes the components of an adaptive strategy "PROG". Each strategy has four main parts: *description*, *variable declarations*, *initialization* and *implementation*. The description is just a comment for the human reader (the author who has to decide to apply this strategy or not). The variables are a new addition, to prevent overlaps and clashes if multiple strategies are applied on the same course: the use of these same variables should be informative about the possible clash. The two phases of initialization and implementation are also new. The *initialization* should set all the variables in use during the strategy, before the actual interaction of the strategy with the user (learner). It also establishes what learning items have to be shown to the user from the very beginning. The *implementation* part contains the actual user interaction, activity description. Initialization and implementation are built from statements. These building blocks are the basis of the current version of the LAG language. The adaptation language also allows assembly language statements, such as IF-THEN statements. However, it also contains more general programming statements, such as WHILE, FOR, and BREAK statements, and comments. The most specific statements are the SPECIALIZE and GENERALIZE statements, that allow the user to go down, or up the learning item hierarchy respectively – depending upon the fulfilment of certain conditions. These statements use the structure of the learning material, therefore have greater semantics for authors familiar with the learning material. The conditions are either prerequisites, or combinations of ENOUGH prerequisites. The value in the latter construct is a number, establishing how many of the prerequisites have to be fulfilled[4]. In such a way, more complex AND-OR combinations of conditions can be obtained. The details of the grammar have been simplified a little. However, it is important to remark that the ATTRIBUTES used in initializations, actions and comparisons can be of two main types: GENERAL or SPECIFIC. The specific attributes refer to an instance of the learning material, whereas the general attributes refer to materials of a given type. Therefore, strategies can be written general enough to be able to be applied to any given set of learning materials, given the condition that these materials can be identified as being of a given type.

From the strategy classifications in section 1.4, LAG can create both *strategies* and *meta-strategies*, as will be shown in the following section. From the point of view of actions, LAG supports *selection*, *showing of content of an item*, *hierarchical actions*, *actions on group*s (except for ordering) and *actions on the overall environment*. *Ordering* is part of the Goal and Constraints Model in LAOS, and *links to items* can only be displayed if they are represented in the Domain or Goal and Constraints Models.

2.2 (Authoring of) Learning styles with LAG

The LAG grammar was used as a basis of the MOT-adapt interface [28]. This interface is depicted in Figures 2 & 3.

Figure 2 shows the *adaptive strategy* written in LAG for the *Verbalizers versus Imagers* learning style, in a Web environment. In terms of presentation mode, the verbalizers should be presented with more textual information, whilst the imagers should be receive more graphic information, such as pictures, diagrams, charts. The value of the VERBvsIM attribute is an integer between 0 and 100. We use a value between 30 and 70 to indicate that the learning style is unknown or the learner has no strong preference. Values between 70 and 100 indicate that the user is a *verbalizer*, values between 0 and 30 indicate that he is *imager*. The strategy is written in a simple way, by using IF-THEN constructs only, to enable an easy comparison with the new XML adaptation language presented later on in the paper (section 3).

Showing the content of an item is done in LAG by an action statement 'PM.Concept.item=true'[5], e.g., 'PM.Concept.image= true' means showing the image. User model attributes are accessed in a similar way: 'UM.Concept.VERBvsIM <30' means that the user tends to be an imager. The strategy is detailed for imagers only, the other case being symmetrical.

[4] The idea behind it is simple yet semantically significant: it is based on computer games, where a player has to collect a number of items to advance between levels. This number may be fixed, but the choice of which items to select is up to him.

[5] PM stands for *Presentation Model* in LAOS.

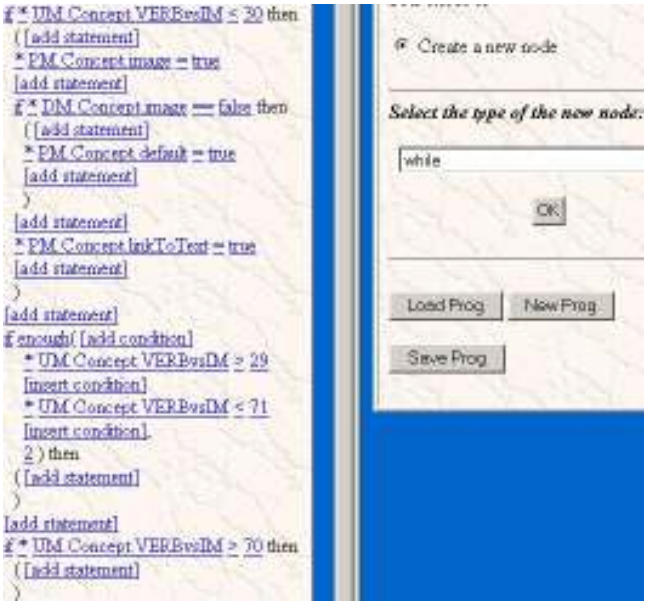

Figure 2. The LAG Grammar: Imager Strategy.

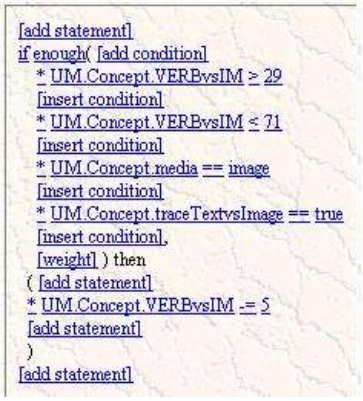

Figure 3. The LAG Grammar: Imager versus Textual Meta-strategy.

-

Figure 3 shows an example of an *adaptive meta-strategy* written based on LAG. The top of the figure also shows the description of the meta-strategy, as defined in Figure 1. The adaptation language constructs and variables are similar to the ones in Figure 2.

Concluding, we can say that LAG allows reusable dynamic representations at different levels: at adaptation language level, by reusing the language constructs, and at adaptation strategy level, by reusing adaptive procedures as new language constructs, but also by reusing whole adaptive strategies (by applying them to different domain maps and user maps, or exporting them to other systems).

3. A New XML Learning Style Adaptation Language and Grammar: LAG-XLS

The new XML language started with the purpose of taking over these advantages of dynamic reuse, whilst adding new research results, as presented in section 1.3: the review on the most frequently used instructional methods to support learning styles. LAG-XLS instantiates the Adaptation Language layer of the LAG model as well, but with different goals. In our new XML-based adaptation language we try to express the first two methods: selection of media items (or selection of a particular type of information in general) and ordering information – in a simple and straightforward manner. Moreover, the refined classification of actions is used as shown in Table 1.

We have based our new language on LAG, and have tried to alleviate some of its problems, whilst at the same time simplifying parts of it. This is based on our desire to identify more specific language constructs aimed at learning style strategies, as well as being completely AHA! compatible. We initially decided to create an XML based language, with the aim of aligning it with semantic web research [40]. Reusability is achieved in the XML Learning Style Adaptation Language for AHA! by specifying each strategy as a separate XML file. XML (E**X**tensible **M**arkup **L**anguage [39]) is a cross-platform, software and hardware independent tool for representing and transmitting data. XML elements for learning adaptive strategies are not yet defined in the literature, so we endeavoured to invent and describe our own elements.

The language built for AHA! bases *selection* and *ordering* of concepts on the attributes and values of their sub-concepts, as follows. In the hierarchy of concept relationships, sub-concepts (defined as AHA! *object* concepts) are the children of a concept (*object* or *page* concept)[6]. The names of the attributes and their values indicate how these sub-concepts represent the parent concept. For instance, if the media attribute is audio, the sub-concept will represent an audio version of the concept.

Another goal was that of expressing monitoring strategies. To achieve this, the adaptation language for AHA! contains elements specifying user model updates.

The resulting LAG-XLS language for AHA!, corresponding to various strategies (extracted from what was previously implemented in 'adaptation assembly' form only, but also from literature review, and informed by the refined classification in Table 1) are presented in the following DTD (Figure 4). The meaning of the DTD *elements* and *attributes* is explained below:

- *strategy* – is the root element of a file corresponding to a strategy, attribute *name* – the name of the strategy;
- *description* – is the strategy meaning; e.g., the corresponding learner model for which this strategy has been created;
- *if* – a statement to specify if-then-else rules (currently we have only *if* statements within the *strategy* element, however we are thinking about applying other statements as well, like *for*, *while*, etc., as in LAG);
- *condition* – appears within an *if* statement; is a Boolean expression which can contain some user-related information, for example, information about the user's learning style;

[6] In AHA! there can be different types of concepts, e.g., *abstract*, *page* or *object* (fragment) concepts. Abstract concepts do not have a resource associated with it. Page concept can have one or more associated resources. Fragment concepts should be included into pages; they can have multiple resources, however they represent alternative versions of a *part* of a page. These resources are well-formed documents, to be scanned by the AHA! engine for other recursively included objects. Therefore they do not have a header and cannot be viewed separately.

- *then* – an element defining a set of actions to be performed when the *condition* is satisfied;
- *else* – an element defining an alternative set of actions;

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT strategy (description, if*)>
<!ATTLIST strategy name CDATA #REQUIRED>
<!ELEMENT description (#PCDATA)>
<!ELEMENT if (condition, then, else)>
<!ELEMENT condition (#PCDATA)>
<!ELEMENT then (select, sort,
navigationType, action*)>
<!ELEMENT else (select, sort,
navigationType, action*)>
<!ELEMENT select (showContent*,
showContentDefault*, showLink*)>
<!ATTLIST select attributeName CDATA
#REQUIRED>
<!ELEMENT sort (showLink*)>
<!ATTLIST sort attributeName CDATA
#REQUIRED>
<!ELEMENT showContent (#PCDATA)>
<!ELEMENT showContentDefault (#PCDATA)>
<!ELEMENT showLink (#PCDATA)>
<!ELEMENT navigationType (#PCDATA)>
<!ELEMENT action (UMvariable, expression)>
<!ATTLIST action UMupdate CDATA #REQUIRED>
<!ELEMENT UMvariable (#PCDATA)>
```

Figure 4. XML Learning Style Adaptation Language DTD.

The following elements are used to define how the adaptation is performed:

- *select* – selecting a concept representation from a number of existing ones to be included into the final presentation;
- *sort* – sequencing different concept representations depending on the user's learning style, reordering them from most to least relevant.

The "select" and "sort" elements have an attribute "attributeName". The value is provided by the author depending on the aspects of the concepts he wants to include or reorder in the final presentation. For example, we have a concept which has several children representing it via different types of media. All the children concepts have an attribute "media". The value of this attribute for different concepts can be "audio", "video", "text", "image", etc. In the final presentation for various strategies (links to) media items can be explicitly included or not; similarly, (links to) media items can be ordered in different ways:

- *showLink* – showing a link to the concept representation;
- *showContent* – showing the content of the concept representation;
- *showDefaultContent* – showing a default content specified by the author in case no other representation is found for a particular concept;
- *navigationType* – type of the navigational structure:
  - "Breadth-first structure" – presenting the material in breadth-first order;
  - "Depth-first structure" – similar for depth-first order;
- *action* –specifies how the user model is updated; attribute *UMupdate* shows whether it is an absolute or relative update;
- *UMvariable* – indicates which user model variable should be updated, namely which attribute of which concept;
- *expression* – is the value used for user model update.

To exemplify the use of the XML adaptation language, we follow the previous example from section 2.2, and write a strategy for the *verbalizer* versus *imager* learning style. Due to of lack of space we present only the part of the XML file corresponding to the *imager* learning style. To indicate that the user is either a verbalizer or imager we use a similar user model attribute as in the LAG example, section 2.2, "VERBvsIM", for the AHA! "personal" concept[7]. In AHA! the learning styles related attributes of this concept can be initialized via the registration form.

The strategy in Figure 5 uses the XML adaptation language elements: *description*, *select*, *showContent*, *showLink*. It also uses traditional AH elements such as IF-THEN constructs.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE strategiesList SYSTEM "strategy.dtd">
<strategy name="VerbalizerVersusImager">
   <description>Strategy for "Verbal" versus "Visual" style of the
Felder-Silverman Learning Model</description>
   <if>
    <condition>personal.VERBvsIM &lt; 30</condition>
    <then>
     <select attributeName="media">
      <showContent>image</showContent>
      <showContentDefault>default</showContentDefault>
      <showLink>text</showLink>
     </select>
    </then>
   </if>
   <if>
    <condition>personal.VERBvsIM &gt; 29 &amp;&amp;
personal.VERBvsIM &lt; 71</condition>
    <then>…</then>
   </if>
   <if>
    <condition>personal.VERBvsIM &gt; 70</condition>
    <then>…<then>
   </if>
</strategy>
```

Figure 5. Strategy of Verbalizer versus Imager.

The meaning of the strategy is that if the user is an imager (personal.VERBvsIM <30)[8] then, for each concept which can be represented by different media types[9], an "image" representation is included in the presentation. If no "image" representation exists, then the default representation provided by the author is used. The author can also specify that links to other concept representations should be included. In the example a link to a textual representation is inserted using the "showLink" element.

This approach is different from the one presented previously [35]. There, in order to provide this kind of adaptive behaviour we had to repeatedly specify the same adaptation rules for all concepts of the application domain which allowed different representations. By using this new approach, we can specify a certain adaptive behavior once, in one strategy and apply it to the whole domain.

---

[7] a pseudo-concept created when a user first logs into the system, storing user information such as name, login, password. As all concepts in AHA!, it can have arbitrary attributes. It can be used to specify attributes reflecting the learning style.

[8] The 'strange' escape sequences & > and < in the XML file are needed because the XML parser will translate them to &, > and <. Without the escaping the XML parser would *interpret*, instead of *translating* them

[9] children of this concept have an attribute "media"

This approach is based on the LAG language of MOT [29], but is adapted to use current W3C Web technologies (such as XML).

Next, we are going to present a short illustrative example of an instructional meta-strategy, corresponding to the LAG meta-strategy in Figure 3. It also can be used to infer the user's preference for either textual or pictorial information. Here, the author specifies the actions which are performed when the user accesses an AHA! concept (like increasing or decreasing the confidence of the system that the learner has a particular learning style). Here we also present only a part of the XML file indicating a decrease in the confidence of the system that the user is a verbalizer; this corresponds to an increase in the confidence that he/she is an imager.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE strategiesList SYSTEM "strategy.dtd">
  <strategy name="TextvsImagePreference">
    <description>Inferring the preference for textual or pictorial
information</description>
    <if>
      <condition>personal.VERBvsIM.initial &gt; 29 &amp;&amp;
personal.VERBvsIM.initial &lt; 71 &amp;&amp; concept.media
== "image" &amp;&amp; personal.traceTextvsImage</condition>
      <then>
        <action UMupdate="relative">
          <UMvariable>VERBvsIM</UMvariable>
          <expression>var:-5</expression>
        </action>
      </then>
    </if>   …
  </strategy>
```

Figure 6. Meta-Strategy of Verbalizer versus Imager.

In this strategy we use two new AHA! variables: *personal.VERBvsIM.initial* and *personal.traceTextvsImage*. Such variables can be added by the author of an application and initialized through the registration form. The first variable stores the initial value of the "VERBvsIM" attribute. The second variable indicates whether the user wants the system to infer his preferences. For example, the user does not know what his learning style is and wants the system to trace it. He might still let the system trace his preferences even if he explicitly specified what his learning style is. If tracing is desired the value of *personal.traceTextvsImage* is set to true. During the actual interaction of the learner with the system, the user's repetitive accesses to pictorial representations increase the confidence of the system that the user is a *imager*, indicated by the expression var:-5. *Var* means that the value can be changed by the author while applying the strategy to a particular application. In the strategy presented in the Figure 6, the default is –5. The system will trace the user's behavior until the value of the "VERBvsIM" attribute reaches a meaningful threshold (30 or 70), then the value of the attribute personal.traceTextvsImage will be set to false and tracing will stop. Afterwards an instructional strategy corresponding to the new value of the "VERBvsIM" will be suggested to the user.

If the learner is not satisfied with an instructional strategy he can always inspect his user model and make necessary corrections. AHA! provides a special tool that allows authors to create forms to let the learners change values of attributes of concepts in their user model. It is thus possible to create a form that lets a learner change their "VERBvsIM" value.

This is an example of an XML adaptation strategy which can be reused by various authors. For their own applications, authors might create their own visions of the verbalizer versus imager strategy or the strategy for tracing the learner's preference for textual or pictorial information. They might use a different attribute of different type indicating the user's style (instead of "VERBvsIM"), they might also specify a different range of values for the attribute and different kinds of adaptation using "showLink", "showContent" elements. They might specify as well a different set of actions for inferring the learner's preferences. The only limitation is that when creating strategies they can only use the elements which are present in the DTD.

The adaptation language for creating strategies allows authors to specify generic adaptation rules. Moreover, the default values of the parameters in each rule can be replaced by the author. Similarly to the MOT adaptation language, the XML adaptation language can deal with specific as well as generic concepts. The examples presented so far only show dealing with generic concepts, specified by the variable "concept". While applying the strategy to an application this name will be replaced with the specific concept names. Moreover, the specific concepts can be directly used in the strategy, as in the example below:

NameSpecificConcept.Attribute = Value

Currently a more user friendly authoring tool for creating adaptive strategies is under development. An authoring tool will allow the authors to create their strategies using the predefined set of elements (specified in the DTD). The authors of adaptive applications should first use the tool to create adaptive strategies – the result will be a separate server-side XML file for each strategy. Skilled authors can also manually create or edit XML files corresponding to strategies, as shown in the examples. The created files will be put into the author's directory, available only for him/herself. A set of "standard" strategies which can be reused by all authors is available and will be extended. If authors want to let others use some of their strategies, they would have to add them to the list of standard strategies.

4. Applying XML Learning Style Adaptation Language to AHA!

In order to visualize the strategies not only from the author's, but also from the delivery (learner's) point of view, let us have a look at how the "VerbalizerVersusImager" strategy (Figure 5) will be converted for the AHA! delivery engine. The author can create the Domain and Adaptation model for the AHA! courses using a high-level authoring tool called Graph Author [15]. A new option has been added to the tool, allowing an author to choose which strategies to apply to a particular course, and in which order (in case of application of several strategies, order can be important). The author might need to rewrite some information (such as the parameters specified with "var"). Otherwise, default values will be applied. During saving, the AHA! concept relationships graph is translated into AHA! low-level (assembly) adaptation rules. The applied strategies may influence the requirements for concepts (the desirability of concepts) and the set of actions to be performed when the concepts are accessed. Additional application pages (in XHTML format) might be generated as well.

The strategy has to be applied to all AHA! concepts in the given course which have sub-concepts with an attribute "media". The application of the XML adaptation strategy in Figure 5 should have as effect the display of the content of the appropriate sub-concept, depending on the value of the attribute ("image",

"default" or "text"), or only a link to inappropriate sub-concepts. For the imager an "image" should be included into the presentation. If an "image" is not found then the system will look for a "default". Due to the fact that inappropriate sub-concepts are added as links, the learner can still follow a link to a "text".

In Figure 7 we show part of the structure of an AHA! parent concept via a simplified syntax. This parent concept represents the conversion of an AHA! concept from the XML adaptive strategy language (Figure 6) to the AHA! low-level assembly language, which the AHA! system can deliver. This example shows that the value of the "VERBvsIM" attribute of the concept "personal" influences the "showability" attribute, which in turn determines the fragment to show.

```
<concept>
 <name>conceptname</name><expr>true</expr>
 <attribute><name>access</name>
 <action>
  <if> personal.VERBvsIM &lt; 30</if>
  <then>conceptname.showability := 0</then>
 </action>
 <action>
  <if> personal.VERBvsIM &gt; 70</if>
  <then>conceptname.showability := 1</then>
 </action> …
 </attribute>…
 <attribute><name>showability</name>
  <casegroup>
  <defaultfragment> generatedfile4.xhtml</defaultfragment>
  <casevalue><value>0</value>
    <returnfragment>generatedfile2.xhtml</returnfragment>
  </casevalue>
  <casevalue><value>1</value>
    <returnfragment>generatedfile3.xhtml</returnfragment>
  </casevalue>
  </casegroup>
 </attribute>
</concept>
```

Figure 7. Example part of the generated structure for the AHA! concept.

The contents of files generatedfile2.xhtml and generatedfile3.xhtml are explained below. They are needed because the conversion into AHA! does not run as smoothly as expected. The "text" concept is an AHA! object concept. Resources associated with this type of concepts can only be seen if included into pages. Therefore, a new page resource file (e.g., generatedfile1.xhtml) that includes it has to be generated, representing a viewable version of the "text" concept, as follows:

```
<!DOCTYPE html SYSTEM "/aha/AHAstandard/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><body>
<object name="objectText" type="aha/text" />
</body></html>
```

The goal of this resource file is to add a header wrapper to the AHA! object concept. The resource file uses an "object" tag for conditional inclusion of objects. The specified type "aha/text" does not mean that the object is a text; it can be any media item. It is used only as an indication that the object should be processed by the AHA! engine.

Afterwards, a resource representing the AHA! parent concept has to be also generated – a page resource, if the AHA! parent concept is a page concept; or a fragment, if it is an object concept. The first case is that of *adaptive link destinations*: i.e., when the learner follows a link to a parent concept, the displayed content of the concept will vary with the user model state. This means that the same link to a concept will point to different resources depending on the user model. The second case results in *adaptation of the content*. This happens if the parent concept is a part of some other page. This page will contain different contents, depending again on the user model. If we assume that the parent concept is an object, the file (generatedfile2.xhtml) corresponds to the AHA! parent concept as follows:

```
<span><object name="objectImage" type="aha/text" /><br />
<a href="generatedfile1.xhtml">Text</a></span>
```

Similarly, the resources representing the parent concept under other conditions (e.g., when the user is a verbalizer or his/her style is not known) will be generated (generated3.xhtml and generatedfile4.xhtml respectively).

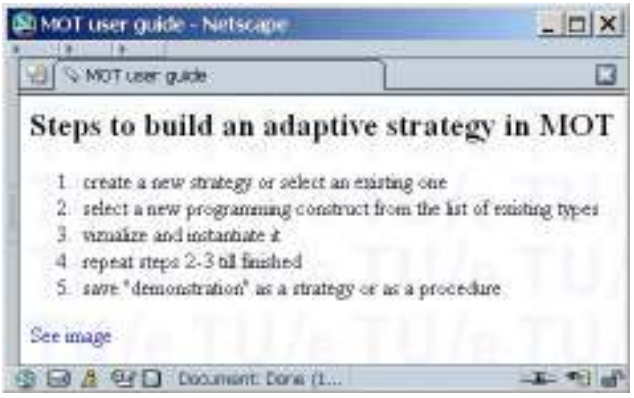

Figure 8. Presentation of MOT user guide to verbalizer.

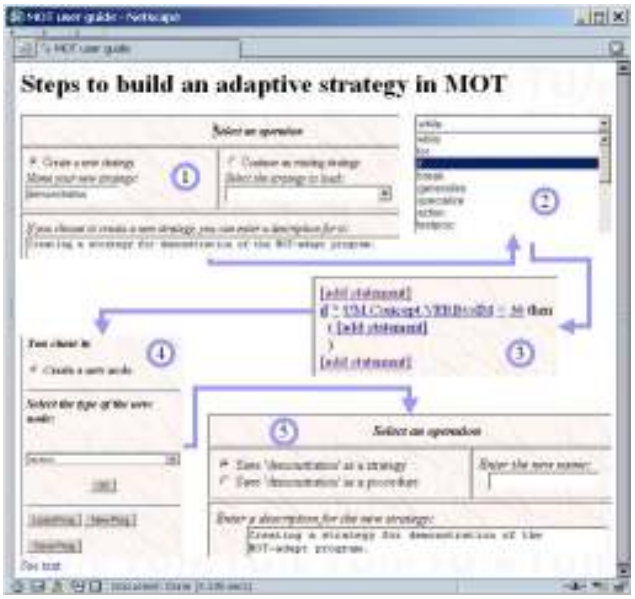

Figure 9. Presentation of MOT user guide to imager.

The figures 8 & 9 illustrate the alternatives for the strategy *visualizer versus imager* delivered by the AHA! Web-based system. The content example is taken from the MOT-adapt user guide for adaptation language creation. Figure 8 presents the textual description of "steps to build an adaptive strategy in MOT" to verbalizers. Correspondingly, Figure 9 presents these steps in a diagram form for imagers.

5. DISCUSSION & CONCLUSION

Before we conclude on the two adaptive languages extracting artificial intelligence features of AH, as described in this paper, we first analyze the few comparable approaches that we have found in the literature. Recently, similar attempts at defining a reusable representation for the system 'intelligence' and dynamics of web-based adaptive education environments have been researched and can be classified into the following categories, as follows:

- *adaptation languages*: In [5], the authors define adaptive rules based on a collection of sets employing the IMS Learning Design [24]. These rules are only at the level of assembly language of adaptation (according to the classification in [12]), i.e., IF-THEN rules, but are enriched with extra semantics. For this, they use semantically labelled actions (such as show, hide, show-menu, sort-ascending, number-to-select, etc.). One problem with this approach is that it mixes the user adaptation (such as some material being *not recommendable* for a user) with the actual presentation of this adaptation (*hide it* from user). This problem is inherited from the strict adherence to the IMS-LD standard, which does not make this distinction. In the adaptive hypermedia literature [6], however, the presentation of an item which is undesirable can vary from *hiding* to *color-code marking* (e.g., 'Red' is undesirable). This type of presentation depends on the degree of control the learner can have within the learning environment. Moreover, the IMS-LD standard is especially aimed at collaboration, and not at personalization.
- *workflow models*: The COW platform in [38] as well as the WFMS in [8] use workflow modeling for dynamics representation. However, in COW no personalization or adaptation is envisioned. WFMS has a form of non-flexible adaptation, comparable with the conditional fragment inclusion technique in early adaptive hypermedia [6].
- *task composition models*: In [7], tasks are modeled and alternative paths are created via AND and OR relations. This alternation seems to be more dynamic than the Simple Sequencing Protocol [25]. The problem is that the language used for task definition is very domain dependent.

LAG has already addressed many of these problems, as it is a higher level language that allows for an increased level of semantics. User adaptation and presentation are kept separate. The adaptivity allowed is extremely flexible and the language is not domain dependent. One important drawback is that it does not reflect the current Web-standards. LAG has been evaluated in real life settings and the results have been described in [11].

The newly proposed LAG-XLS adaptation language is aimed at alleviating this last problem. XML is a universally accepted standard way of structuring data, where XML web-designers are not restricted to a limited set of tags. Therefore the adaptation language can be created and extended by defining new tags. The language makes use of the new classification of actions (Table 1). Moreover, the XML syntax ensures Web-readability and the capacity to export to different systems. LAG-XLS has been evaluated in real life settings and the results reported in [36].

The focus of the new language is however slightly different from LAG, which is a more generic adaptation language, as the new language specifically targets users' learning styles and the adaptive strategies corresponding to them, restricted by the DTD definitions.

Currently we have defined and are experimenting in LAG-XLS with a number of *instructional* strategies other than the one represented in this paper, such as Active versus Reflective, Auditory versus Visual, Holist versus Analytic, Field-Dependent versus Field-Independent, Verbal versus Visual learners as well as other *monitoring* meta-strategies, like inferring preferences for textual or pictorial information or reading in breadth- or depth-first order. We are thinking about the extension of the adaptation language for defining more complex variations of the strategies. We are planning to apply OWL (Web Ontology Language) as it provides a number of useful constructs "oneOf", "intersectionOf", "unionOf", etc.

Both LAG and LAG-XLS instantiate the LAG Model adaptation language. Both languages respond to the main solution envision here: *reuse*. This paper therefore demonstrates that separating the specific dynamics required for the complex issue of learning style adaptive response is possible, and therefore paves the way for exportable, reusable adaptive strategies on a global scale and their integration into Web standards. We have demonstrated this by comparing two adaptation languages, starting with what problems they solve, what their underlying model is, how they differ from other approaches, and what their positive and negative aspects are.

Moreover, by making the 'intelligence' in the adaptive hypermedia systems explicit, not only can these AH systems be analyzed as to the extent of 'intelligence' they can represent; but also, in this way, the adaptive model is only weakly connected to the delivery engine, and can therefore be easily replaced with other alternative approaches of machine intelligence representation, such as fuzzy logics, neural networks, etc. In this way, the artificial intelligence part of the AH systems is clearly delimited and defined, and plug-and-play technology becomes applicable. Existing educational hypermedia can therefore be reused in new, adaptive & intelligent ways – however more research is necessary for establishing the requirements of merging at both syntactic and semantic levels.

6. ACKNOWLEDGMENTS

This work is supported by the NLnet Foundation and by the ADAPT project (101144-CP-1-2002-NL-MINERVA-MPP).

7. REFERENCES

[1] Abdullah, N. A. and Davis, H. C. Is Simple Sequencing Simple Adaptive Hypermedia?. In Proceedings of Hypertext 2003, Nottingham, 172-173.

[2] AHA!. http://aha.win.tue.nl.

[3] ADL, SCORM, http://www.adlnet.org/index.cfm?fuseaction=scormabt

[4] ARIADNE, Foundation for the European Knowledge Pool, http://www.ariadne-eu.org/

[5] Berlanga, A. and Garcia, F.J., Towards Reusable Adaptive Rules, Workshop on AH and Collaborative Web-based Systems, ICWE'04.

[6] Brusilovsky, P. Adaptive hypermedia. User Modeling and User Adapted Interaction, 11(1/2), (2001), 87-110.

[7] Carro, R.M., Moriyón, R., Pulido, E. and Rodríguez, P. (1999): Teaching Tasks in an Adaptive Learning

Environment. In: HCI Communication, Cooperation and Application Design, Eds: Bullinger, H. and Ziegler, J., Vol 2, 740-744.

[8] Cesarini, M., Monga , M., Tedesco, R. Carrying on the e-learning process with a workflow management engine, Proceedings of the 2004 ACM symposium on Applied computing, March 14-17, 2004, Nicosia, Cyprus.

[9] Coffield, F., Learning Styles and Pedagody in post-16 learning: A systematic and critical review. Learning & Skills research centre. http://www.lsda.org.uk/files/pdf/1543.pdf

[10] Conlan, O., Lewis, D. Higel, S., O'Sullivan, D. and Wade, V., Applying Adaptive Hypermedia Techniques to Semantic Web Service Composition Twelfth International World Wide Web Conference, Budapest, Hungary, May 20, 2003.

[11] Cristea, A and Cristea, P. Evaluation of Adaptive Hypermedia Authoring Patterns During a Socrates Programme Class, Journal of Advanced Technology for Learning, 1(2), ACTA Press, 2004.

[12] Cristea, A.I., and Calvi, L. The three Layers of Adaptation Granularity. UM'03. Springer.

[13] Cristea, A., De Mooij, A. LAOS: Layered WWW AHS Authoring Model and its corresponding Algebraic Operators. In Proceedings of WWW'03, Alternate Education track. (Budapest, Hungary 20-24 May 2003). ACM.

[14] Dagger, D., Developing Adaptive Pedagogy with the Adaptive Course Construction Toolkit (ACCT), Second International Workshop on Authoring of Adaptive and Adaptable Educational Hypermedia, AH'04, http://wwwis.win.tue.nl/~acristea/AH04/workshopAH.htm

[15] De Bra, P., Aerts, A., Rousseau, B., Concept Relationship Types for AHA! 2.0, *Proceedings of the AACE ELearn'2002 conference*, Montréal, Canada, 2002, 1386-1389.

[16] de La Passardiere, B. & Dufresne, A.: Adaptive navigational tools for educational hypermedia. In: Tomek, I. (ed.) Computer Assisted Learning. Springer-Verlag, Berlin,1992, 555-567.

[17] Dolog, P., Henze, N., Nejdl, W. and Sintek, M. The Personal Reader: Personalizing and Enriching Learning Resources using Semantic Web Technologies, In AH 2004.

[18] Dublin Core Metadate Initiative. http://dublincore.org

[19] EML http://eml.ou.nl/eml-ou-nl.htm

[20] Felder, R.M. & Soloman, B.A. (2000). Learning styles and strategies.http://www2.ncsu.edu/unity/lockers/users /f/felder/public/ILSdir/styles.html

[21] IEEE P1484.2/D7, 20000-11-28. Draft Standard for Learning Technology. Public and Private Information (PAPI) for Learners (Papi Learner). http://ltsc.ieee.org/wg2/

[22] IEEE 1484.12.1-2002.1Standard for Learning Object Metadata http://ltsc.ieee.org/wg12/

[23] IMS. IMS LIP (Learner Information Specification). http://www.imsproject.org/profiles/index.cfm

[24] IMS Global Learning Consortium."IMS Learning Design Information Model" (IMS-LD), version 1.0 Final Specification,20 january 2003. http://www.imsproject.org/learningdesign/index.cfm

[25] IMS Global Learning Consortium, "Simple Sequencing Protocol", Version 1.0 Final Specification, March 2003, http://www.imsglobal.org/simplesequencing/index.cfm

[26] iWeaver. http://www.adaptive-learning.net/media/html/iWeaver.htm

[27] Kravcik, M. and Specht, M. "Authoring Adaptive Courses – ALE Approach", 1st International workshop on Adaptive and adaptable Authoring, In: Proc. of the WBE 2004 Conference, Innsbruck, Austria, February 2004.

[28] Moore, A.; Brailsford, T.J & Stewart, C.D. (2001). Personally tailored teaching in WHURLE using conditional transclusion. Short Paper, 12th ACM Hypertext Conference. Arhus, Denmark, August 14-18, 2001.

[29] MOT. http://wwwis.win.tue.nl/~acristea/mot.html.

[30] Object Management Group.http://www.omg.org

[31] Peters A. M.. Languages learning strategies: Does the whole equal the sum of the parts? Language (1977), 53, 560-573.

[32] Riding, R.J. & Buckle, C.F. *Learning styles and training performance* (Sheffield: Training Agency, 1990).

[33] Specht, M. and Klemke, R.: ALE- Adaptive Learning Environment. WebNet 2001, 1155-1160.

[34] Stash, N., Cristea, A., De Bra, P. Authoring of Learning Styles in Adaptive Hypermedia: Problems and Solutions. In Proceedings of WWW'04, Alternate Education track. (New York, US 17-22 May 2004). ACM.

[35] Stash, N., De Bra, P., Incorporating Cognitive Styles in AHA! (The Adaptive Hypermedia Architecture), 1st International workshop on Adaptive and adaptable Authoring, WBE'04, Innsbruck, Austria, February 16-18, 2004, 378-383.

[36] Stash, N., Cristea, A., De Bra, P., Empirical Evaluation of a New Adaptation Language for Learning Styles, Intelligence-based Adaptation for Ubiquitous Multimedia Communications, Special issue of the Journal of Network and Computer Applications (to be submitted).

[37] Triantafillou, E., Pomportsis. A, and Georgiadou, E. AES-CS: Adaptive Educational System base on cognitive styles. In AH2002 Workshop (Malaga, Spain, 2002), 10-20.

[38] Vantroys, T and Peter, Y. COW, a Flexible Platform for the Enactment of Learning Scenarios. In. J. Favela and D. Decouchant (eds.) CRIWG 2003, Springer, LNCS 2806, 168-182.

[39] W3C.XML Protocol specification. http://www.w3.org/2000/xp/.

[40] W3C Semantic Web. http://www.w3.org/2001/sw/

[41] Witkin, H.A., Moore, C.A., Goodenough, D.R. & Cox, P.W. Field-dependent and field-independent cognitive styles and their educational implications, *Review of Educational Research, 47*(1), 1977, 1-64.

[42] Wu, H. A. Reference Architecture for Adaptive Hypermedia Applications, doctoral thesis, Eindhoven University of Technology, The Netherlands, ISBN 90-386-0572-2, 2002.